Amendments to the Claims:

This listing of claims will replace all prior versions and listings of claims in the application.

Listing of Claims:

1. (Currently Amended) A system comprised of a computer processor configured for executing a computer program stored in computer memory so as to regulate resource consumption in a computer system used for utility work and production work, the system further comprising:

~~an~~ manager module arrangement for determining and registering at least one utility within the computer system, wherein said registering informs the manager module of the existence of the at least one utility;

an arrangement for deriving a throttling level for the at least one utility which quantifies the reduction in the rate at which the at least one utility consumes resources; and

an arrangement for optionally inserting the derived throttling level at a selected point during execution of the at least one utility, wherein inserting the derived throttling level is updated dynamically through several iterations of a work loop until said utility has completed its work and then deregisters with the manager module;

wherein said arrangement for optionally inserting the derived throttling level is implemented within the at least one utility, said utility being configured to dynamically self-throttle and not require an operating system to throttle the utility.

2. (Previously Presented) The system according to **Claim 1**, wherein said arrangement for determining ascertains whether the at least one utility has indicated its presence with the computer system.

3. (Previously Presented) The system according to **Claim 2**, wherein indicating the presence of the at least one utility within the computer system comprises the at least one utility registering with a utility manager.

4. (Canceled)

5. (Previously Presented) The system according to **Claim 2**, wherein the derived throttling level is enforced through a self-imposed sleep.

6. (Previously Presented) The system according to **Claim 2**, wherein the at least one utility is a multi-process utility and the derived throttling level is enforced by reducing the parallelism of multi-processes.

7. (Previously Presented) The system according to **Claim 2**, wherein the derived throttling level is enforced by reducing the amount of memory used by the at least one utility.

8. (Previously Presented) The system according to **Claim 2**, wherein the derived throttling level is enforced by changing the granularity of locking.

9. (Previously Presented) The system according to **Claim 2**, wherein the derived throttling level is enforced by reducing the amount of processing accomplished by the at least one utility.

10. (Canceled)

11. (Previously Presented) The system according to **Claim 2**, wherein the derived throttling level is enforced by reducing the operating system priority of the at least one utility.

12. (Currently Amended) A method for regulating resource consumption in a computer system used for utility work and production work, the method comprising the steps of:

determining and registering at least one utility within the computer system, wherein said registering informs a manager module of the existence of the at least one utility;

deriving a throttling level for the at least one utility which quantifies the reduction in the rate at which the at least one utility is processed or otherwise consumes resources; and

optionally inserting the derived throttling level at a selected point during execution of the at least one utility, wherein inserting the derived throttling level is updated dynamically through several iterations of a work loop until said utility has completed its work and then deregisters with the manager module;

wherein the derived throttling level is implemented within the at least one utility, said utility being configured to dynamically self-throttle and not require an operating system to throttle the utility.

13. (Previously Presented) The method according to **Claim 12**, wherein said determining step comprises ascertaining whether the at least one utility has indicated its presence with the computer system.

14. (Currently Amended) The method according to **Claim 13**, wherein indicating the presence of the at least one utility within the computer system comprises the at least one utility registering with a utility manager.

15. (Canceled)

16. (Previously Presented) The method according to **Claim 13**, wherein the derived throttling level is enforced through a self-imposed sleep.

17. (Previously Presented) The method according to **Claim 13**, wherein the at least one utility is a multi-process utility and the derived throttling level is enforced by reducing the parallelism of multi-processes.

18. (Previously Presented) The method according to **Claim 13**, wherein the derived throttling level is enforced by reducing the amount of memory used by the at least one utility.

19. (Previously Presented) The method according to **Claim 13**, wherein the derived throttling level is enforced by changing the granularity of locking.

20. (Previously Presented) The method according to **Claim 13**, wherein the derived throttling level is enforced by reducing the amount of processing accomplished by the at least one utility.

21. (Canceled)

22. (Previously Presented) The method according to **Claim 13**, wherein the derived throttling level is enforced by lowering the operating system priority of the at least one utility.

23. (Currently Amended) A program storage device readable by machine, tangibly embodying a program of instructions executable by the machine to perform a method for regulating resource consumption in a computer system used for utility work and production work, the method comprising the steps of:

determining and registering at least one utility within the computer system, wherein said registering informs a manager module of the existence of the at least one utility;

deriving a throttling level for the at least one utility which quantifies the reduction in the rate at which the at least one utility is processed or otherwise consumes resources; and

optionally inserting the derived throttling level at a selected point during execution of the at least one utility, wherein inserting the derived throttling level is updated dynamically through several iterations of a work loop until said utility has completed its work and then deregisters with the manager module;

wherein the derived throttling level is implemented within the at least one utility, said utility being configured to dynamically self-throttle and not require an operating system to throttle the utility.